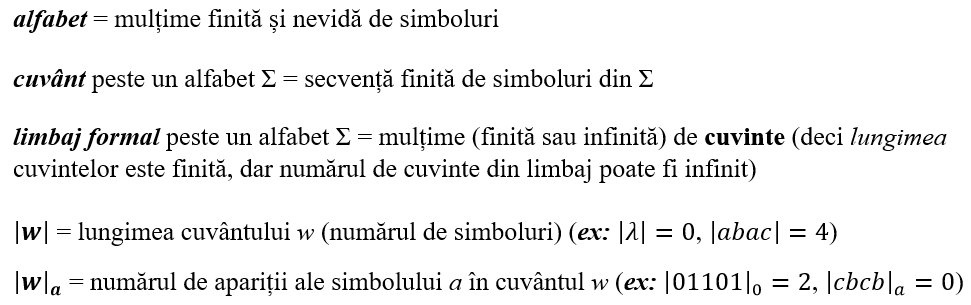
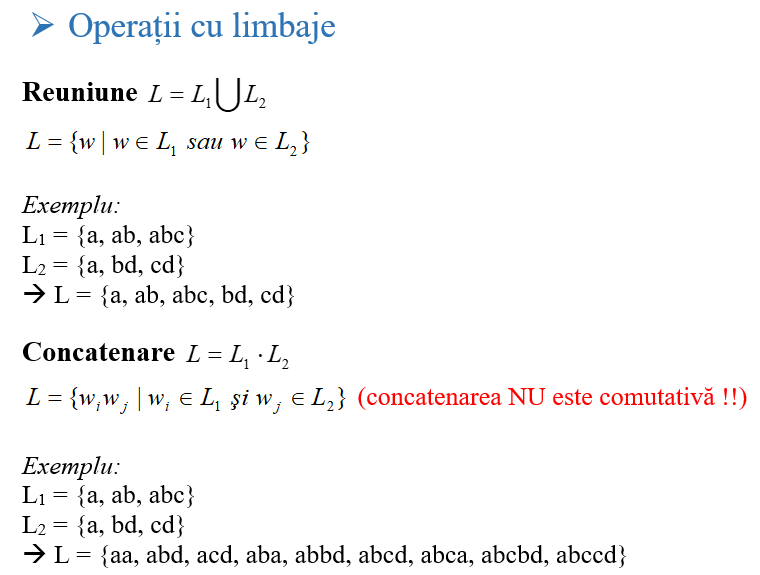
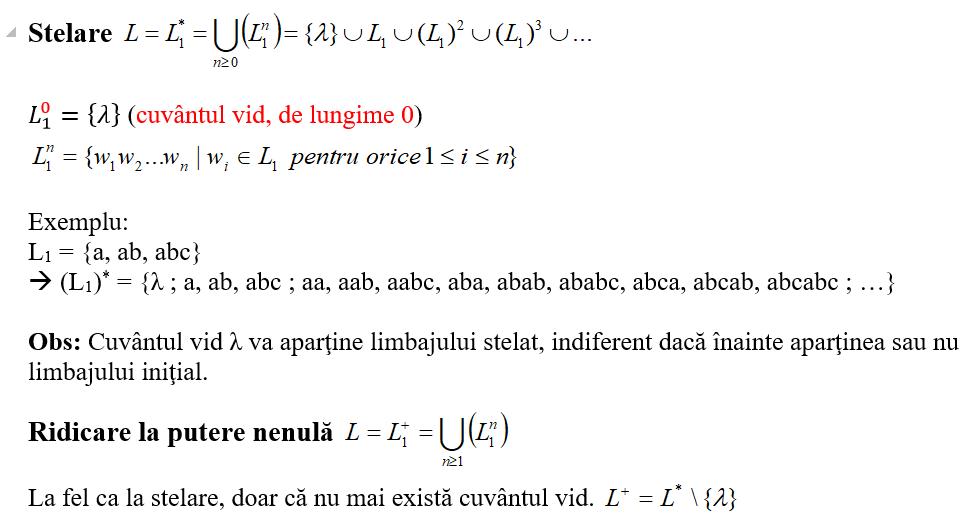
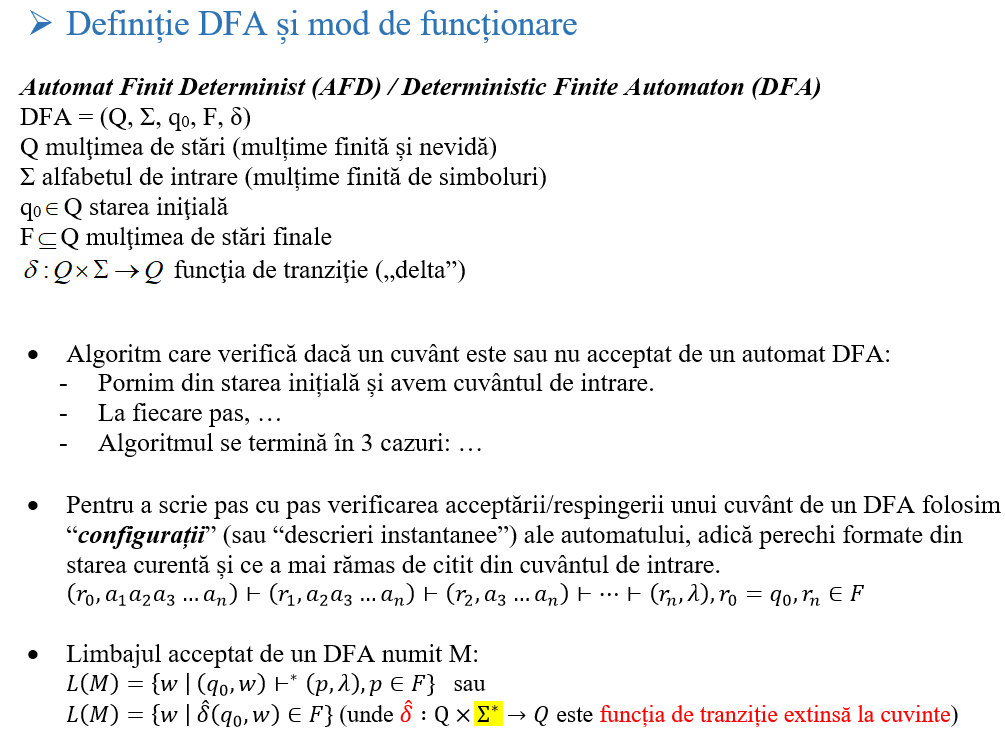
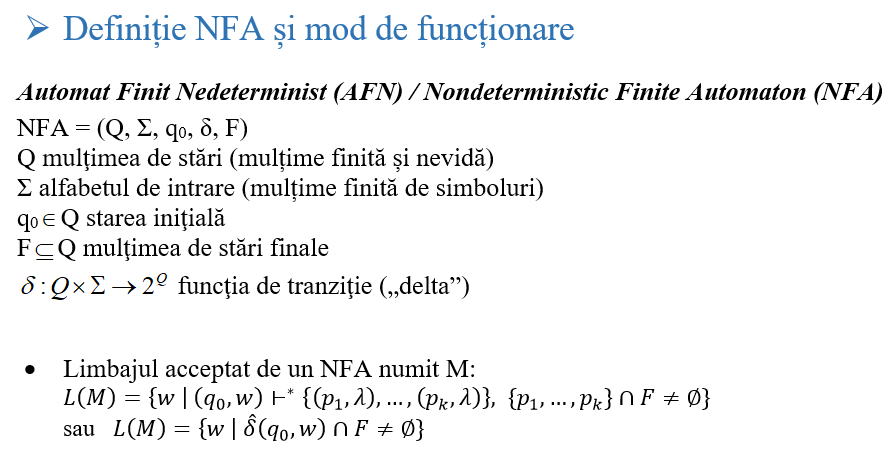
**~ Seminar 1 ~**

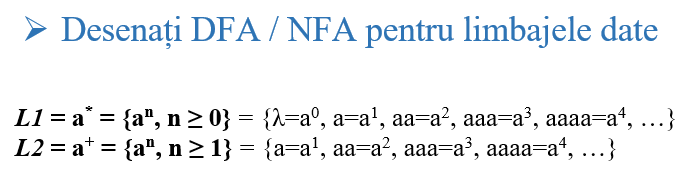


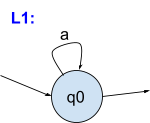


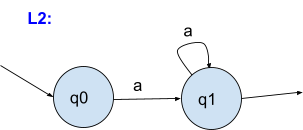


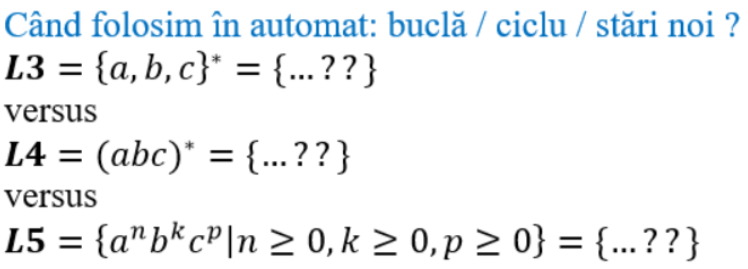




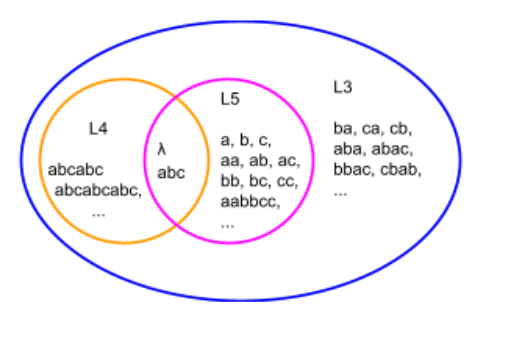


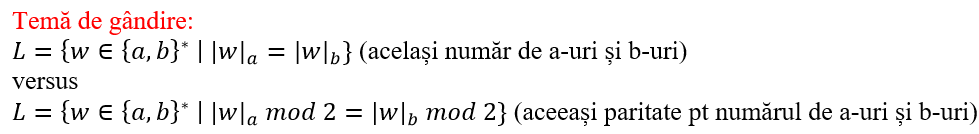




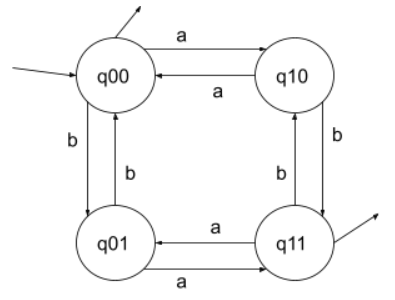


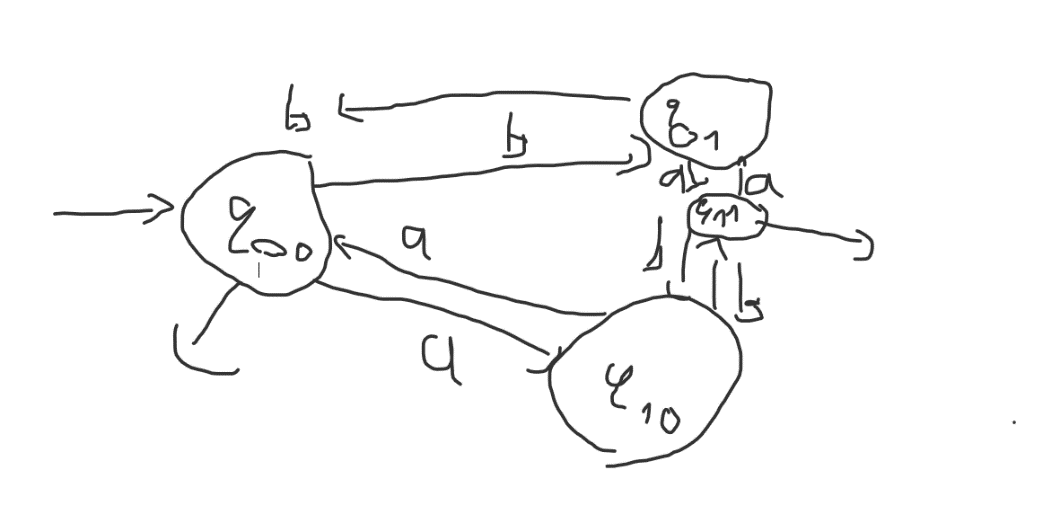
Există vreo relație de incluziune / intersecție între unele dintre limbajele L3, L4, L5 ?



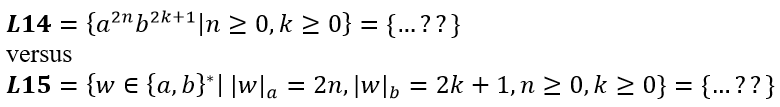


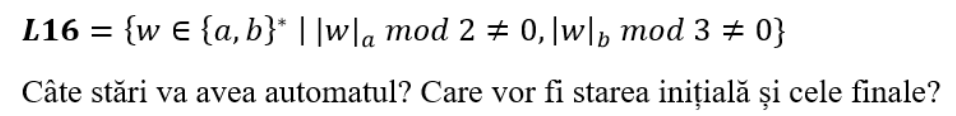
* Putem să desenăm un DFA? Dar un NFA?
* Puteți să dați un exemplu de limbaj pentru care putem desena un NFA, dar nu un DFA?



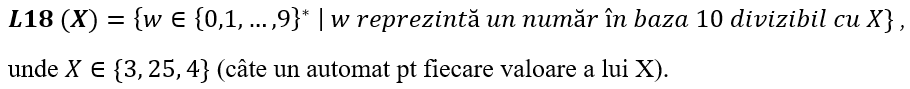


**Temă ??**



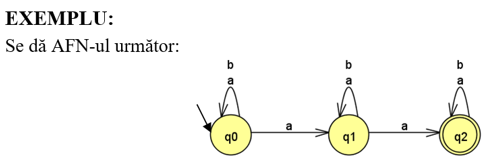


(Adică literele a și b sunt amestecate în cuvânt astfel încât nicio literă nu apare de mai mult de 2 ori consecutiv în cuvânt.)



**~ Seminar 2 ~**



 NFA

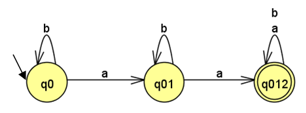
a) Completăm tabel\_1 cu funcția de tranziție pentru AFN.

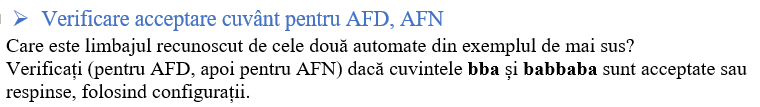
b) Completăm tabel\_2 cu funcția de tranziție pentru AFD (pornim din starea inițială a AFN-ului și adăugăm pe rând stările obținute în interiorul tabel\_2).

c) Desenăm graful pentru AFD conform tabel\_2.

|  |  |  |
| --- | --- | --- |
| **𝜹\_AFN** | **a** | **b** |
| **q0 init** | {q0, q1} = q01 | {q0} |
| **q1** | {q1, q2} = q12 | {q1} |
| **q2 in F** | {q2} | {q2} |

|  |  |  |
| --- | --- | --- |
| **𝜹\_AFD** | **a** | **b** |
| **q0 init** | q01 | q0 |
| **q01** | q012 | q01 |
| **q012 in F** | q012 | q012 |

AFD



--- AFD, cuv bba

(q0, bba) (q0, ba)  (q0, a)  (q01, λ), q01 not in F => bba respins

--- AFN, cuv bba

(q0, bba) (q0, ba)(q0, a){(q0,λ), (q1,λ)}, {q0, q1} intersectat F = mult vida => bba respins

--- AFD, cuv babbaba

(q0, babbaba)(q0, abbaba) (q01, bbaba) (q01, baba) (q01, aba) (q012, ba) (q012, a) (q012, λ), q012 in F => babbaba acceptat

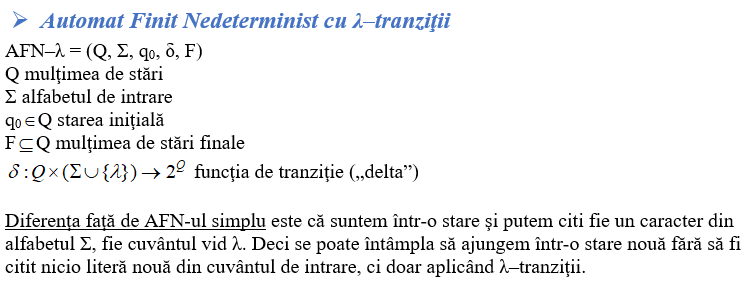
--- AFN, cuv babbaba

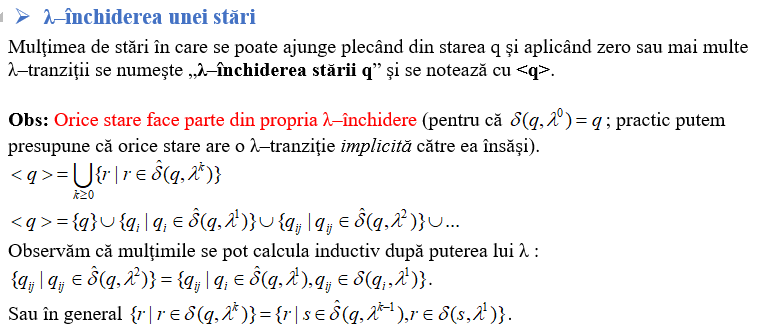
(q0, babbaba) (q0, abbaba) {(q0, bbaba), (q1, bbaba)}

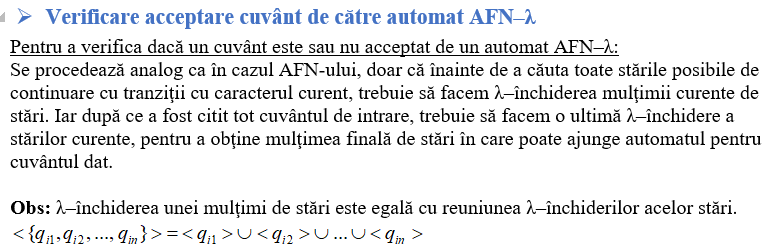
 {(q0, baba), (q1, baba)} {(q0, aba), (q1, aba)} {(q0, ba), (q1, ba), (q2, ba)}

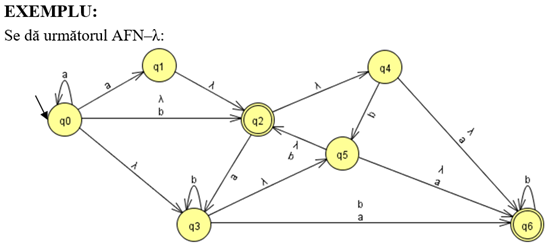
{(q0, a), (q1, a), (q2,a)} {(q0, λ), (q1, λ), (q2, λ)},

 {q0, q1, q2} intersectat F = {q2} (diferit de mult vida) => babbaba acceptat









a) Calculăm λ–închiderile tuturor stărilor.

<q0> = {q0, q2, q3, q4, q5, q6}

<q1> = {q1, q2, q4, q6}

<q2> = {q2, q4, q6}

<q3> = {q3, q5, q2, q6, q4}

<q4> = {q4, q6}

<q5> = {q5, q2, q4, q6}

<q6> = {q6}

b) Verificăm dacă cuvântul **abbaa** este acceptat sau respins de acest AFN-λ, folosind configurații.

(q0, abbaa) (q023456, abbaa) (q0136, bbaa) (q0123456, bbaa) 

(q2365, baa) (q23456, baa)  (q3652, aa) (q23456, aa) 

(q36, a)  (q23456, a)  (q36, λ) (q23456, λ)

{q2, q3, q4, q5, q6} intersectat F = {q2, q6} (diferit de multimea vida) => abbaa acceptat

**EX:** Desenați un AFN / AFD pentru limbajele următoare.



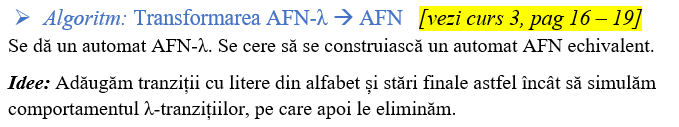
= {a^5, a^8, a^11, a^14, ...}

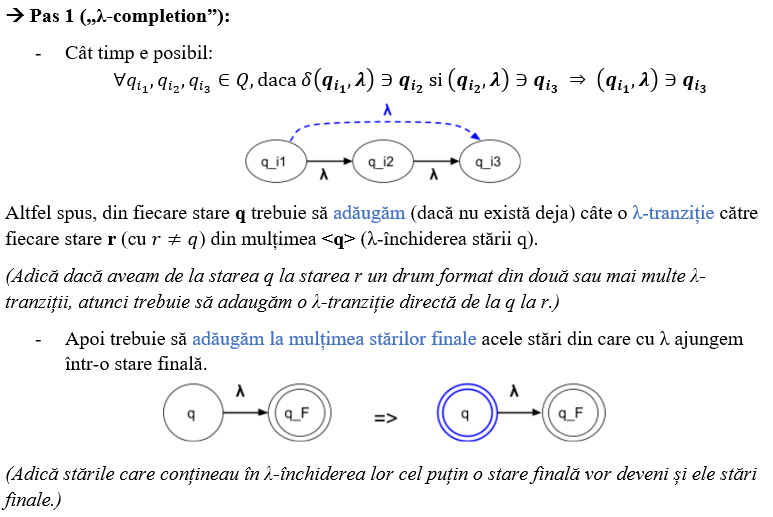


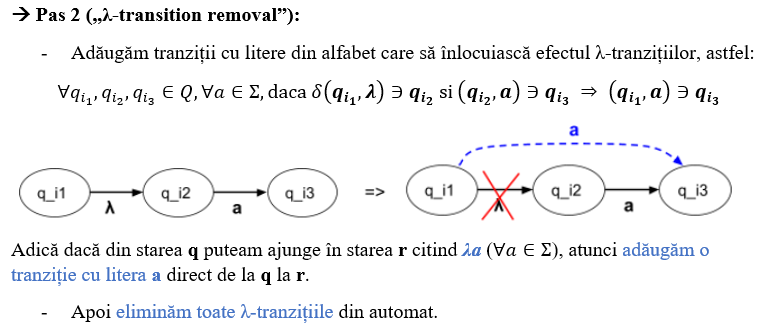




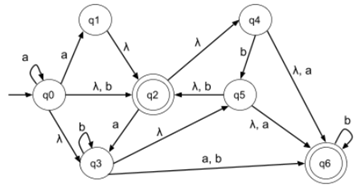
**~ Seminar 3 ~**



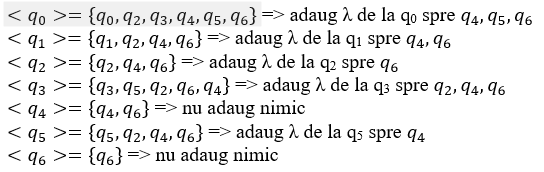




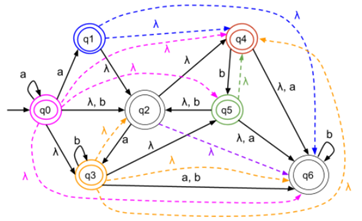




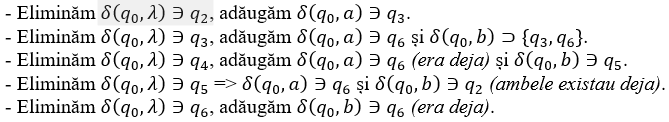
**-> Pas 1 („λ-completion”):**

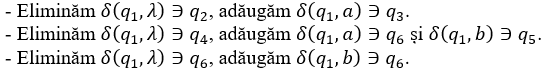


Toate stările au în λ-închiderile lor cel puțin o stare finală, deci toate stările vor deveni finale.

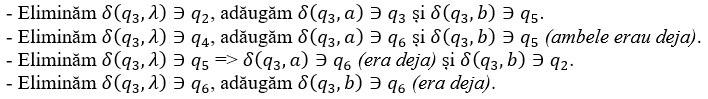
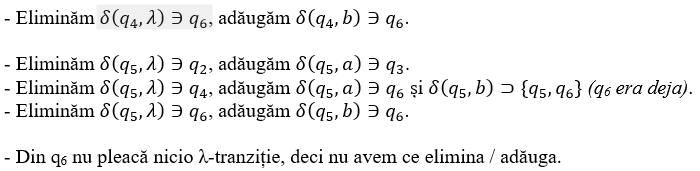


**-> Pas 2 („λ-transition removal”):**



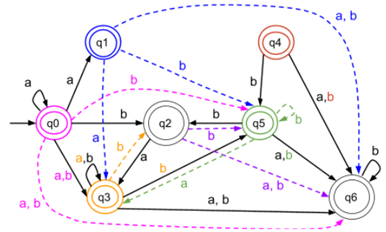


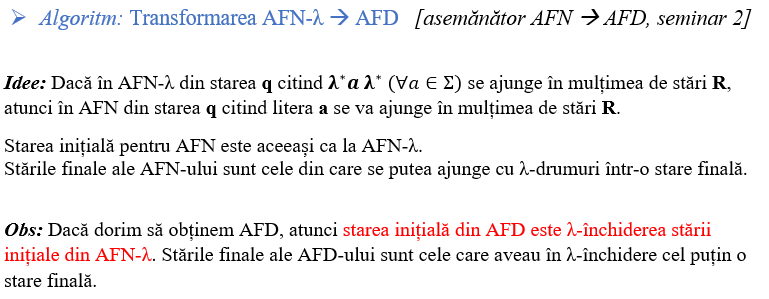




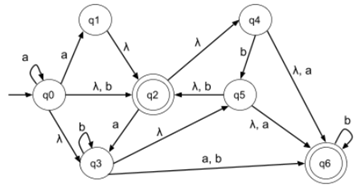
Am obținut un AFN echivalent cu AFN-λ dat.

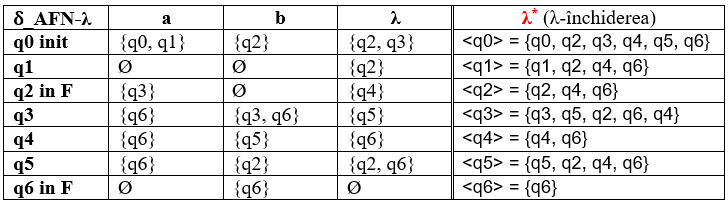
*(Observăm că starea q4 nu este accesibilă din starea inițială, deci ar putea fi eliminată împreună cu trazițiile ei fără a afecta limbajul recunoscut de automat.)*

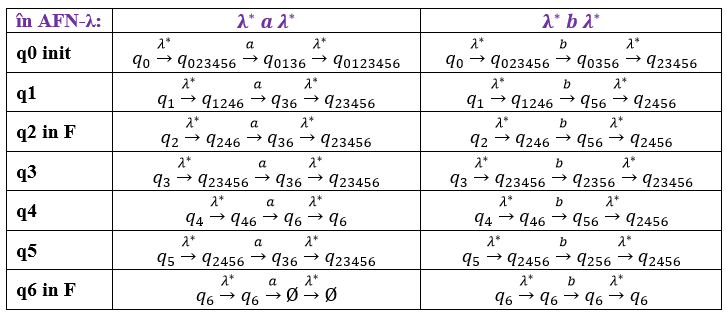


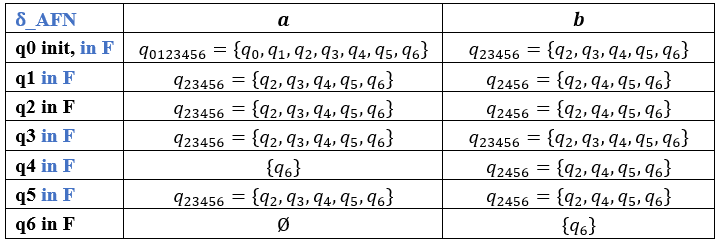


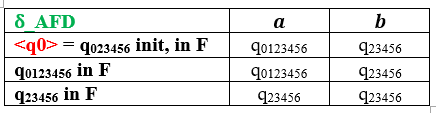






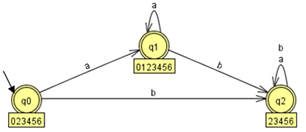


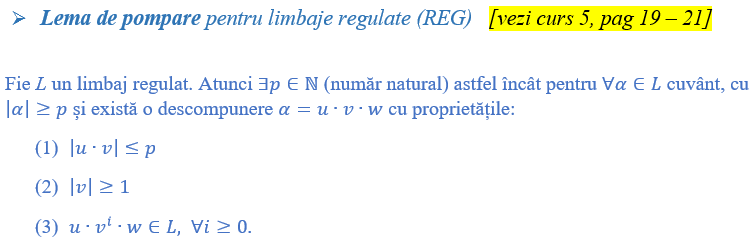


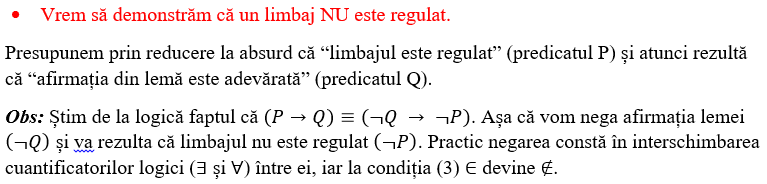


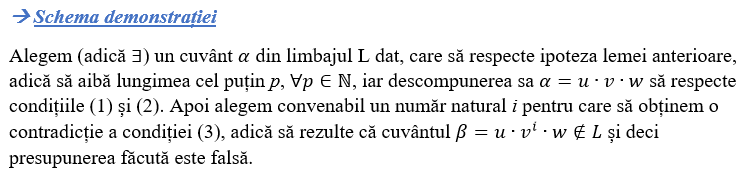
Am obținut un AFD echivalent cu automatele AFN și AFN-λ de mai sus.

Ce limbaj recunoaște acest AFD?









L = {a^m b^n | m > n >= 0} not in REG

***Demonstratie:***

Presupunem prin reducere la absurd ca L este in REG. Rezulta p nr natural din lema.

*(Incepem sa negam lema)*

Alegem alpha = a^(p+1) b^p in L => |alpha| = 2p+1 >= p, pt orice p nr nat.

Cuv alpha se decompune in alpha = uvw a.i. |uv|<=p si |v|>=1 (deci 1<=|v|<=p)

=> uv contin doar a-uri, deci notam v = a^k => |v|=k.

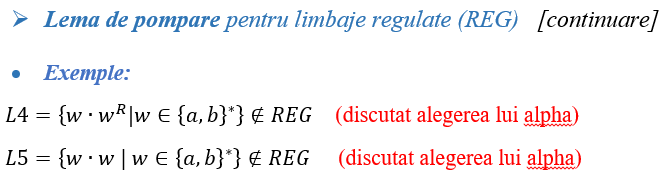
Din cond (1) si (2) => 1 <= k <=p (rel A).

Pt cuv pompat alegem i = 0 => beta = u v^0 w = a^(p+1 - k) b^p in L ⇔

p+1-k > p ⇔ 1 > k contradictie cu (rel A).

Deci L nu e REG.

**~ Seminar 4 ~**



Pt L4, alpha = a^p b b a^p DA

alpha = a^p a^p = a^(2p) = (aa)^p NU e o alegere buna !

Pt L5, alpha = a^p b a^p b



Presupunem prin reducere la absurd ca L6 este REG. Atunci exista p nr natural din lema.

Alegem cuvantul alpha = a^(p^2) in L6 => |alpha| = p^2 >=p, pt orice p nr natural.

Avem alpha = uvw a.i. |uv|<=p si |v|>=1 (deci 1<=|v|<=p).

Notam v = a^k => |v| = k => 1<=k<=p (\*).

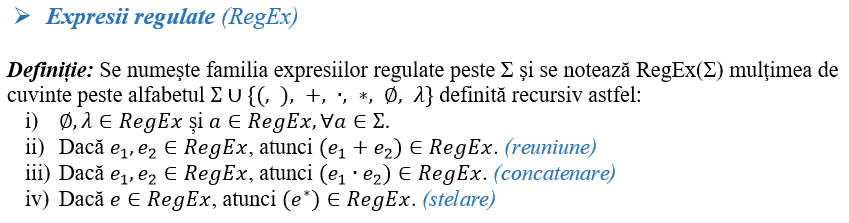
Alegem i = 2 => beta = u v^2 w de lungime

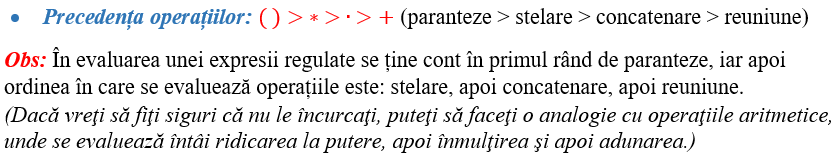
|beta| = |u v^2 w| = |u v w| + |v| = |alpha| + |v| = p^2 + k.

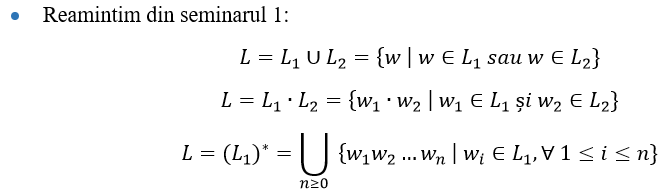
Din (\*) stim ca 1<=k<=p  (adunam p^2) ⇔ p^2 + 1 <= p^2 +k <= p^2 + p

Dar p^2 < p^2+1 si p^2 + p < (p+1)^2

=> p^2 < p^2 + k < (p+1)^2 ⇔ p^2 < |beta| < (p+1)^2  => deci |beta| nu poate fi patrat perfect => beta nu apartine L6, contradictie cu conditia (3) din lema => L6 nu e in REG.









Pentru fiecare caz din definiția RegEx vom construi câte un automat finit echivalent.

**Caz i)**

|  |  |  |  |
| --- | --- | --- | --- |
| **RegEx** | e=∅ | e= | e=a, unde a |
| **Limbaj** | L=∅ | L={} | L={a} |
| **Automat Finit** |  |  |  |

În **cazurile ii), iii) și iv)** presupunem că

pentru expresia regulată ek și limbajul L(ek), k{1,2} avem deja

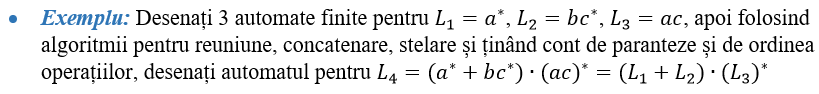
automate finite AF(L(ek)) = (Qk, k ,q0k,Fk,k), cu Q1Q2=∅ (stări disjuncte).

Desenăm schema unui automat punând în evidență starea inițială q0k și mulțimea stărilor finale Fk. Dreptunghiul Mk include toate celelalte stări și tranzițiile automatului.

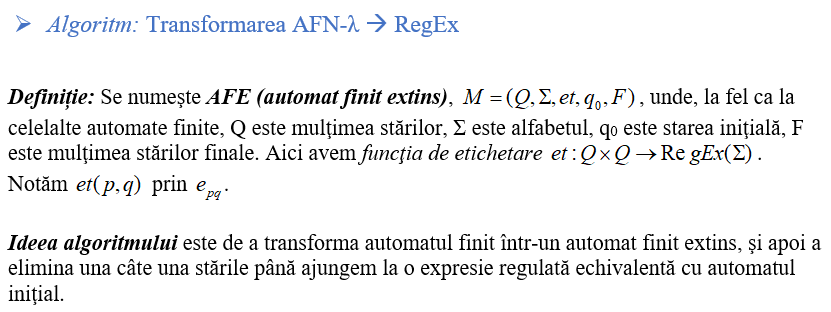


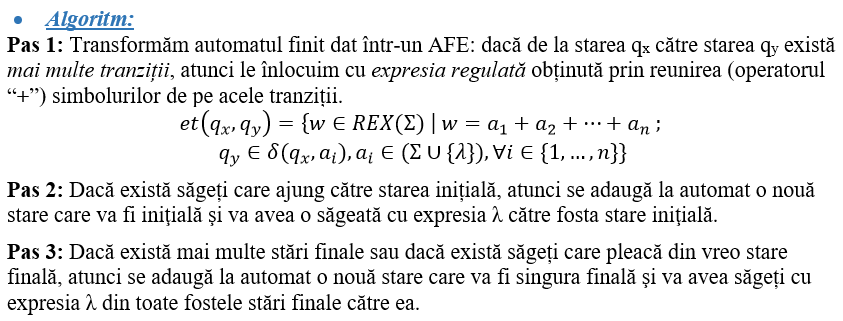
Vom construi automatele pentru operațiile de reuniune, concatenare și stelare.

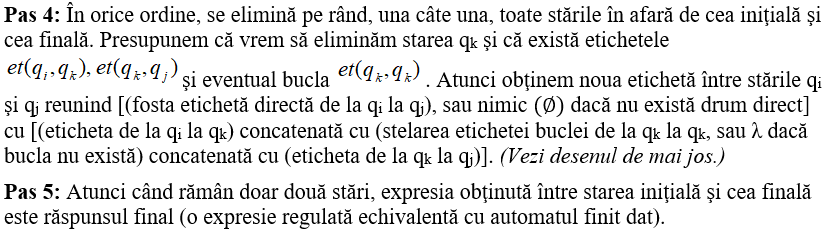
|  |  |
| --- | --- |
| **Caz ii)**  *(reuniune)*  **RegEx**  e=e1+e2  **Limbaj**  L(e)=L(e1)L(e2) | **Automat Finit**  AF(L(e1+e2)) = (Q1Q2{q0}, 12, q0, F1F2,  12{(q0,)={q01,q02}}) |
| **Caz iii)**  *(concatenare)*  **RegEx**  e=e1e2  **Limbaj**  L(e)=L(e1)L(e2) | **Automat Finit**  AF(L(e1e2)) = (Q1Q2, 12, q01, F2,  12{(qf1,)q02 | f1F1}) |
| **Caz iv)**  *(stelare)*  **RegEx**  e=(e1)\*  **Limbaj**  L(e)=(L(e1))\* | **Automat Finit**  AF(L((e1)\*)) = (Q1{q0}, 1, q0, F1{q0},  1{(q0,)=q01}{(qf1,)q01 | f1F1}) |

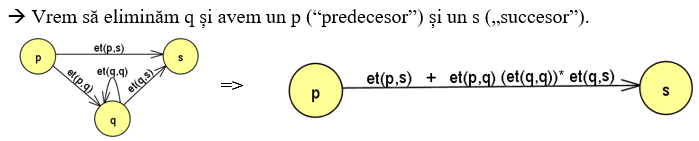


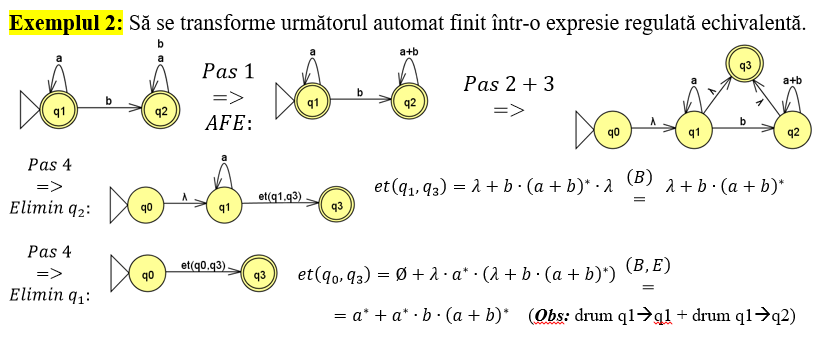
*(desen pe whiteboard)*

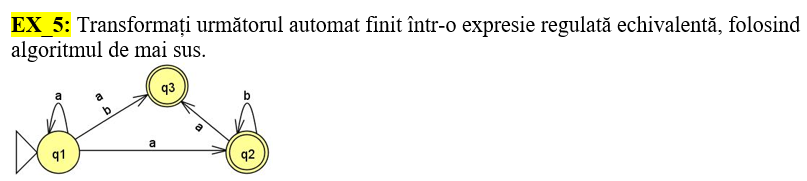




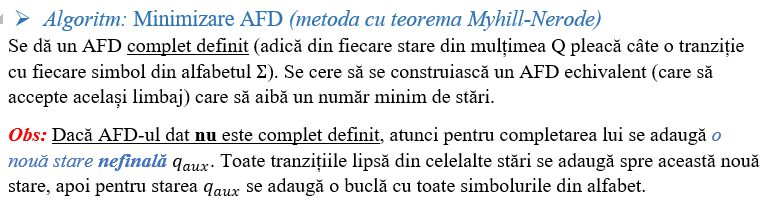


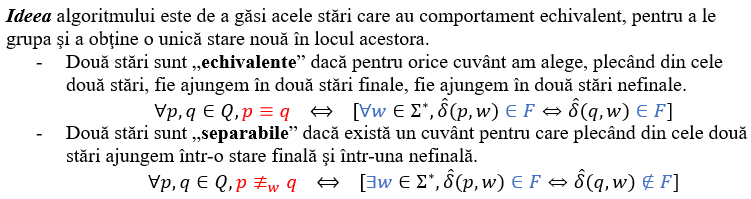


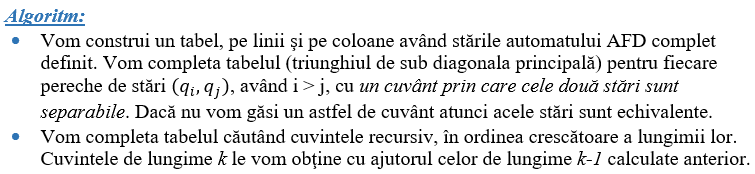




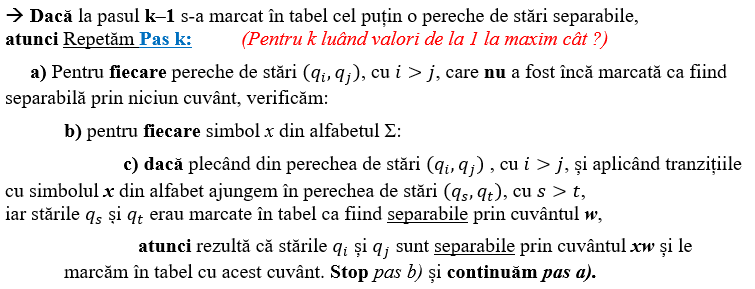
**~ Seminar 5 ~**

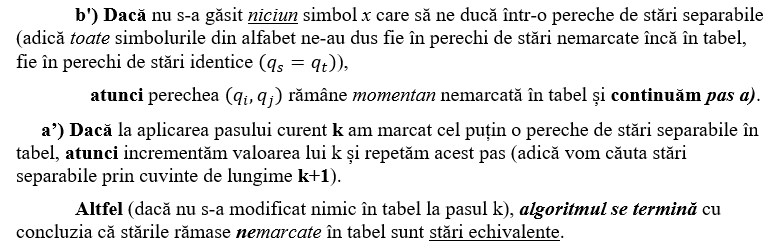


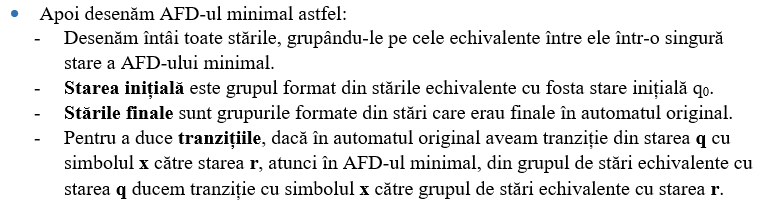


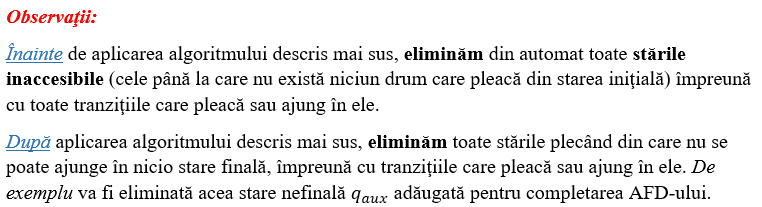


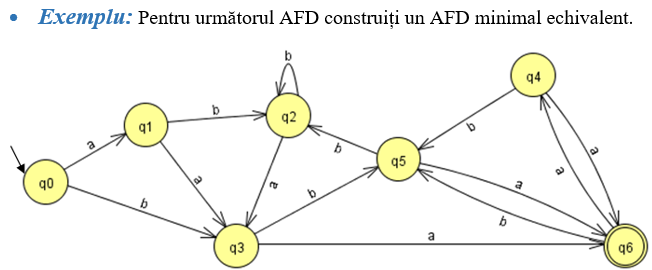












Observăm că AFD-ul dat este complet definit şi nu există stări inaccesibile.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | q0 | q1 | q2 | q3 | q4 | q5 | q6 |
| q0 |  |  |  |  |  |  |  |
| q1 | aa |  |  |  |  |  |  |
| q2 | aa | mult vida |  |  |  |  |  |
| q3 | a | a | a |  |  |  |  |
| q4 | a | a | a | mult vida |  |  |  |
| q5 | a | a | a | ba | ba |  |  |
| q6 | λ | λ | λ | λ | λ | λ |  |

